MASTER OF SCIENCE IN ENGINEERING

Multimodal Processing, Recognition & Interaction

Laboratoire 1 - ANN

Favre-Bulle Matthieu - Mueller Michael

2015

HES-SO//Master
Orientation TIC

Professeur	: J. Hennebert
Branche	: MPRI
Salle de laboratoire	: 5
Dernière mise à jour	: 9 mars 2015

Table des matières

Chapitre 1

Introduction

Le but de ce laboratoire était des découvrir une des méthodes possible de reconnaissance multi-modale : les réseaux de neurones. L'objectif final était d'entrainer un réseau tel que celui-ci, afin qu'il puisse différencier notre voix des autres étudiants de la classe.

Nous avons longtemps travaillé sur le sujet afin d'obtenir des résultats satisfaisants. Toutefois, les solutions ne sont pas toujours parfaites, voir incomplète par manque de temps.

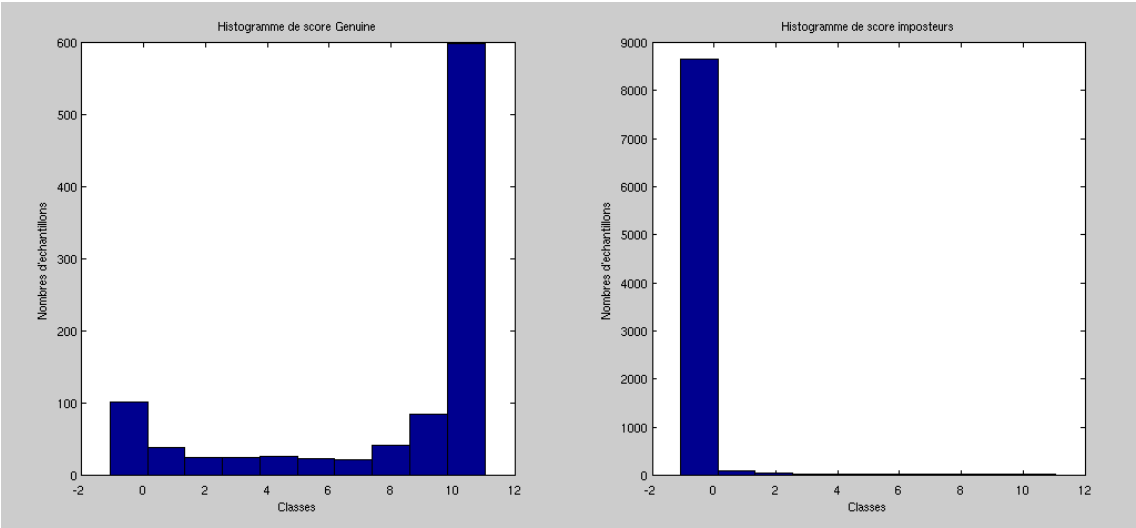
Chapitre 2

Biometric System Evaluation

2.1 Scores distribution

Le but de cet exercice est d'analyser les performance d'un système biométrique à 2 classes. Ce système possède deux listes de scores : "genuine" et "impostors".

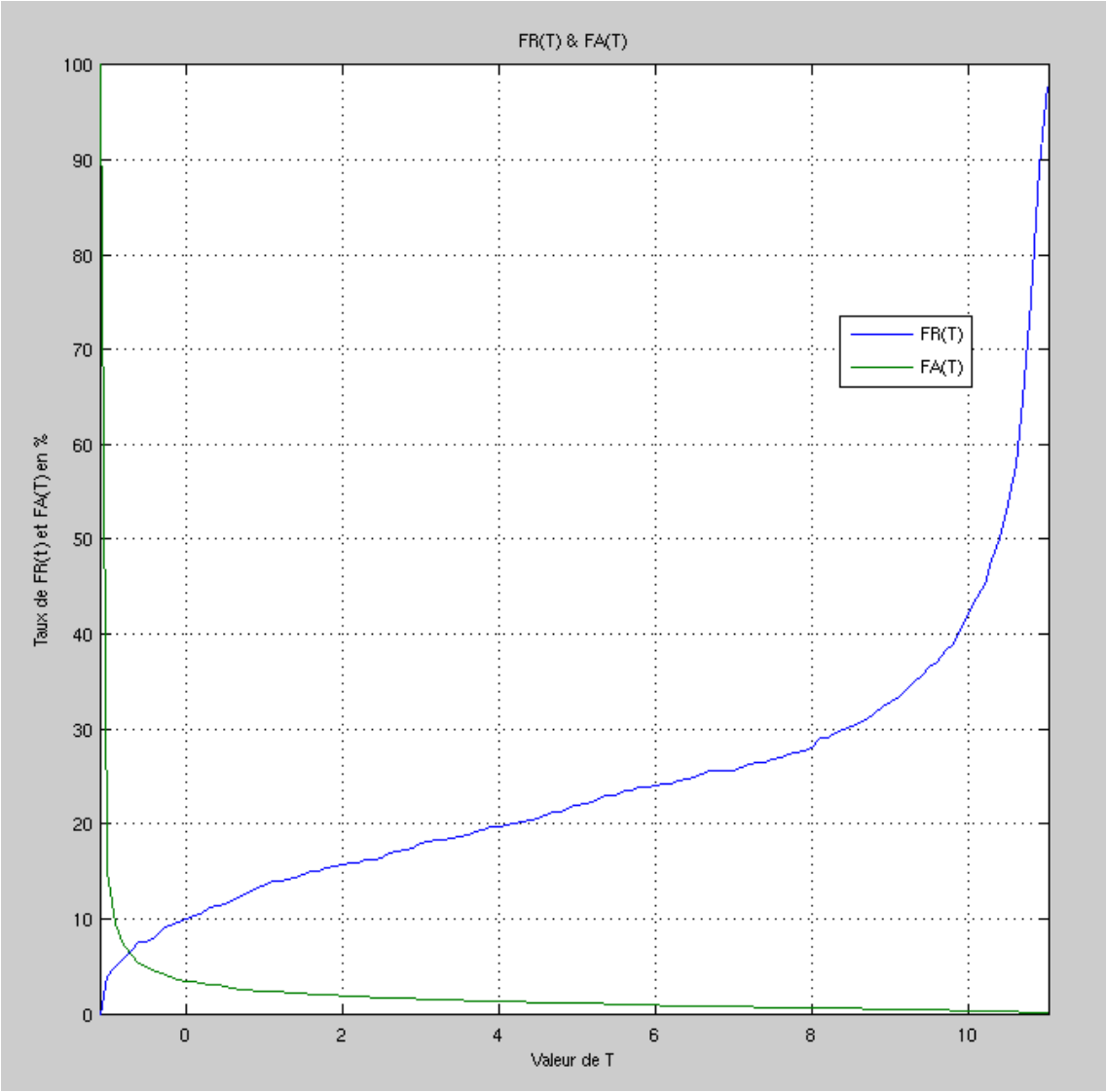
Voilà le résulat de la fonction "hist()" mise en forme avec matlab :



On peut remarquer le score "genuine" (à gauche) et score "impostors" (à droite) ont une répartitions des scores bien distincte. Toutefois, on peut remarquer qu'il a un petit recouvrement des zones. Cela signifie que l'on aura probablement des quelques fausse réjections ou même fausses acceptations selon le seuil appliqué.

2.2 Biometric performance curves

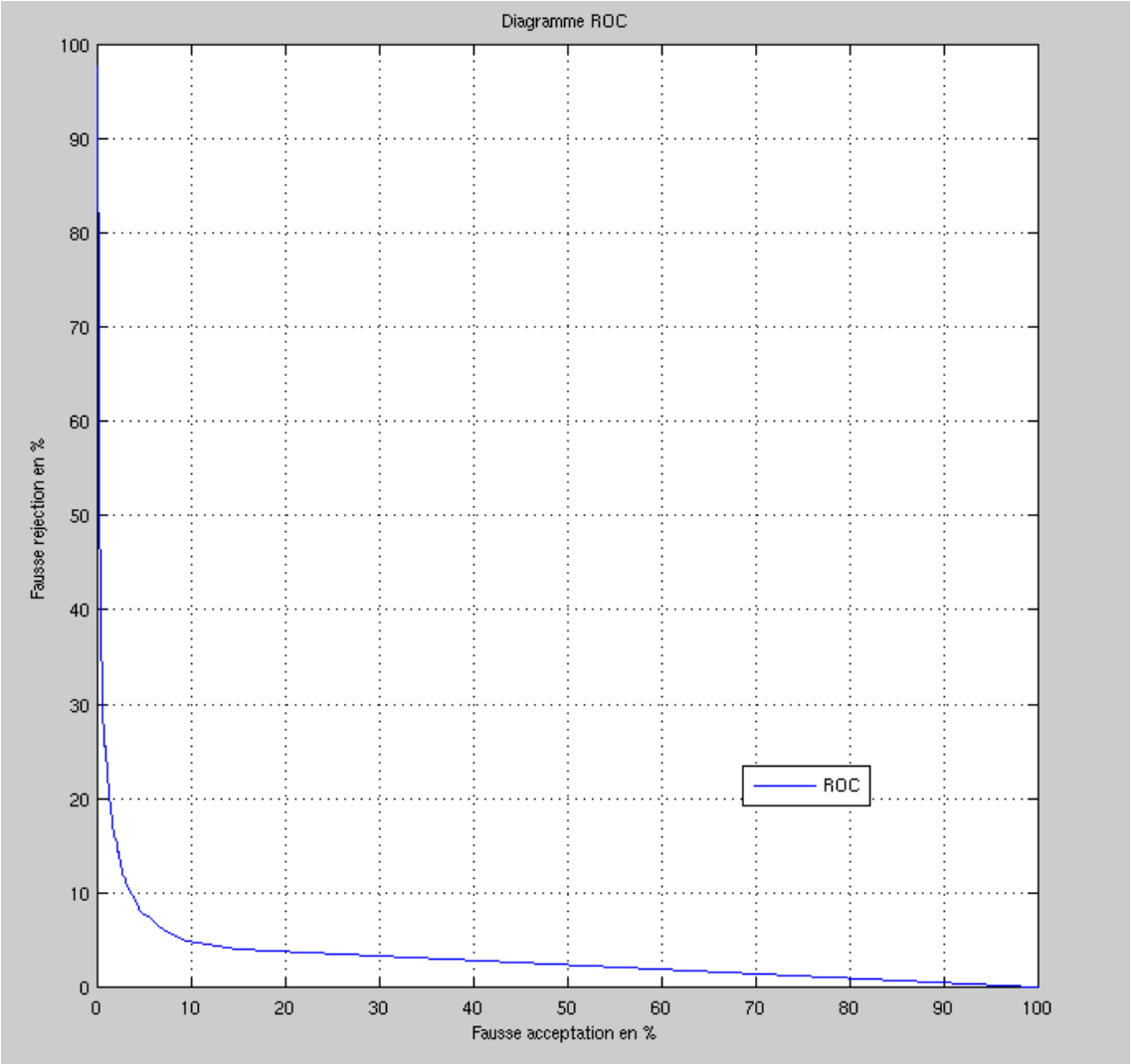
Cette partie de cet exercice consistait à tracer différents graphique de performance du système biométrique étudié.
Voilà les courbes FA(T) et FR(T) :



On remarque (même que c'est assez logique) que plus "T" est petit, plus le taus de fausses acceptations augmente (on accepte tout le monde). On voit aussi que plus notre "T" est grand, plus on aura tendance à refuser le vrai utilisateurs du système. Le but est d'essayer de trouver un bon compromis, entre refuser un utilisateur ou accepter un imposteur.

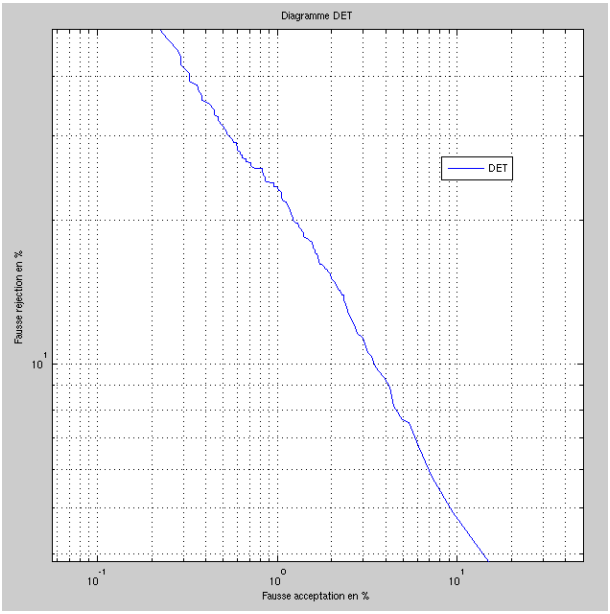
Remarques : On peut observer un recouvrement des deux courbes, ce que l'on avait aussi observé sur les histogrammes.

Voilà maintenant la courbe ROC linéaire :

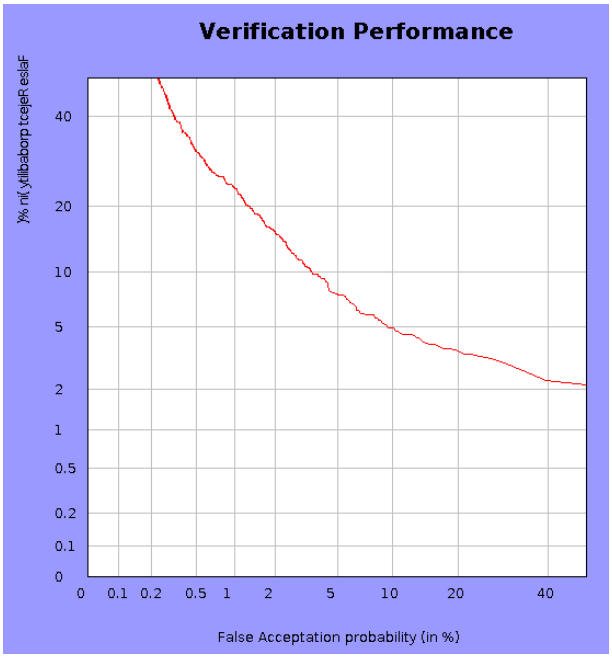


Cette fois on à représenté le taux probable de fausses réjections et fausses acceptations. Pus le coude est proche de l'origine, meilleur sera notre système.

Pour terminer, voilà la courbe ROC logarithmique, nommée DET :



Nous n'avons malheureusement pas compris pourquoi cette courbe ne ressemble pas à celle proposé par le petit logiciel Java fourni sur "Cyberlearn". Pourtant les autre courbes, elles paraissent juste ! Voici ce que nous aurions du obtenir :



Peut-être serait-ce uniquement une histoire d'échelle ? Nous ne savons pas... Le fait est que nous nous sommes arrêtés ici pour cet exercice (par manque de résultats et de temps).

Chapitre 3

ANN to estimate a cosine function

Le but de cet exercice est de démontrer que l'on peut apprendre "n'importe quoi" à un réseau de neurone. Pour notre essai, nous allons apprendre à notre réseau de neurones la fonction :

$$f(x) = cos(x)$$

3.1 Training data

Pour commencer nous allons générer une suite de valeur avec le "target" correspondant (jargon matlab). Voilà le code utilisé :

```
1 x_train = [0:0.1:2*pi] ;
2 y_target = cos(x_train);
```

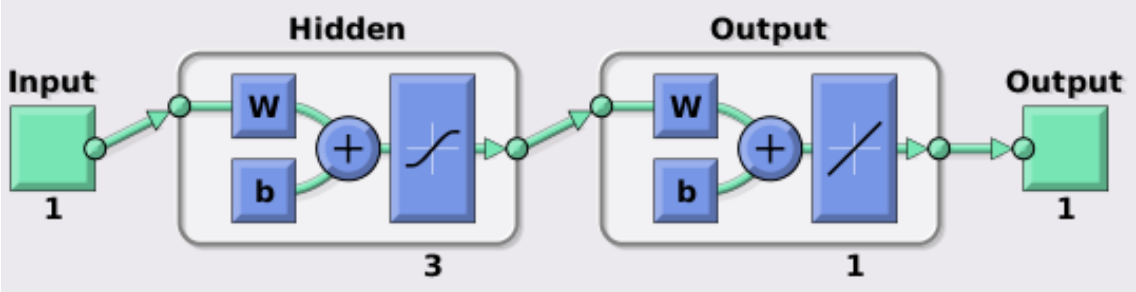
On va générer une matrice $x_{train} \in \{0, 0.1, ..., 2\pi\}$ et une matrice de résultat $y_{target} = cos(x_{train})$. Ces données vont nous permettre d'entrainer notre réseau de neurones.

3.2 ANN training

Voilà la déclaration de notre réseau de neurones (nous avons fait l'essai avec 3 layers et 10 layers pour comparer les résultats).

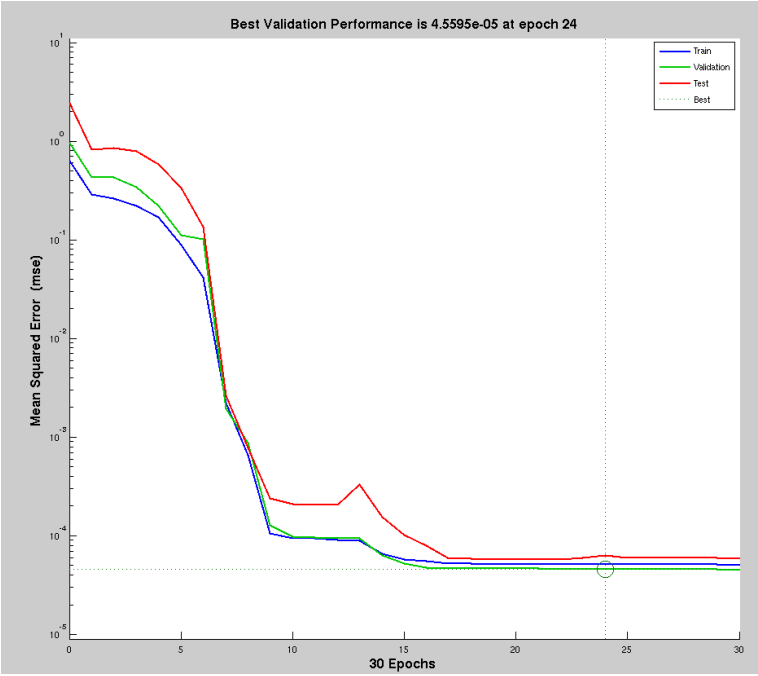
```
net = feedforwardnet(3);
net10 = feedforwardnet(10);

net = train(net,x_train,y_target);
net10 = train(net10,x_train,y_target);
```

Voilà la représentation de notre MLP à 3 layers selon l'outil matlab (nous n0avons pas bien compris pour quoi le dernier étage est différent des 3 autres. Probablement pour adapter la sortie) :



On voit que l'on à qu'une entrée et une sortie, ce qui est logique, car on veut entrer une valeur de "x" et obtenir le cosinus de "x". Le type d'entrainement utilisé est "Levenberg-Marquardt".

Voici le diagramme MSE généré par notre outils :



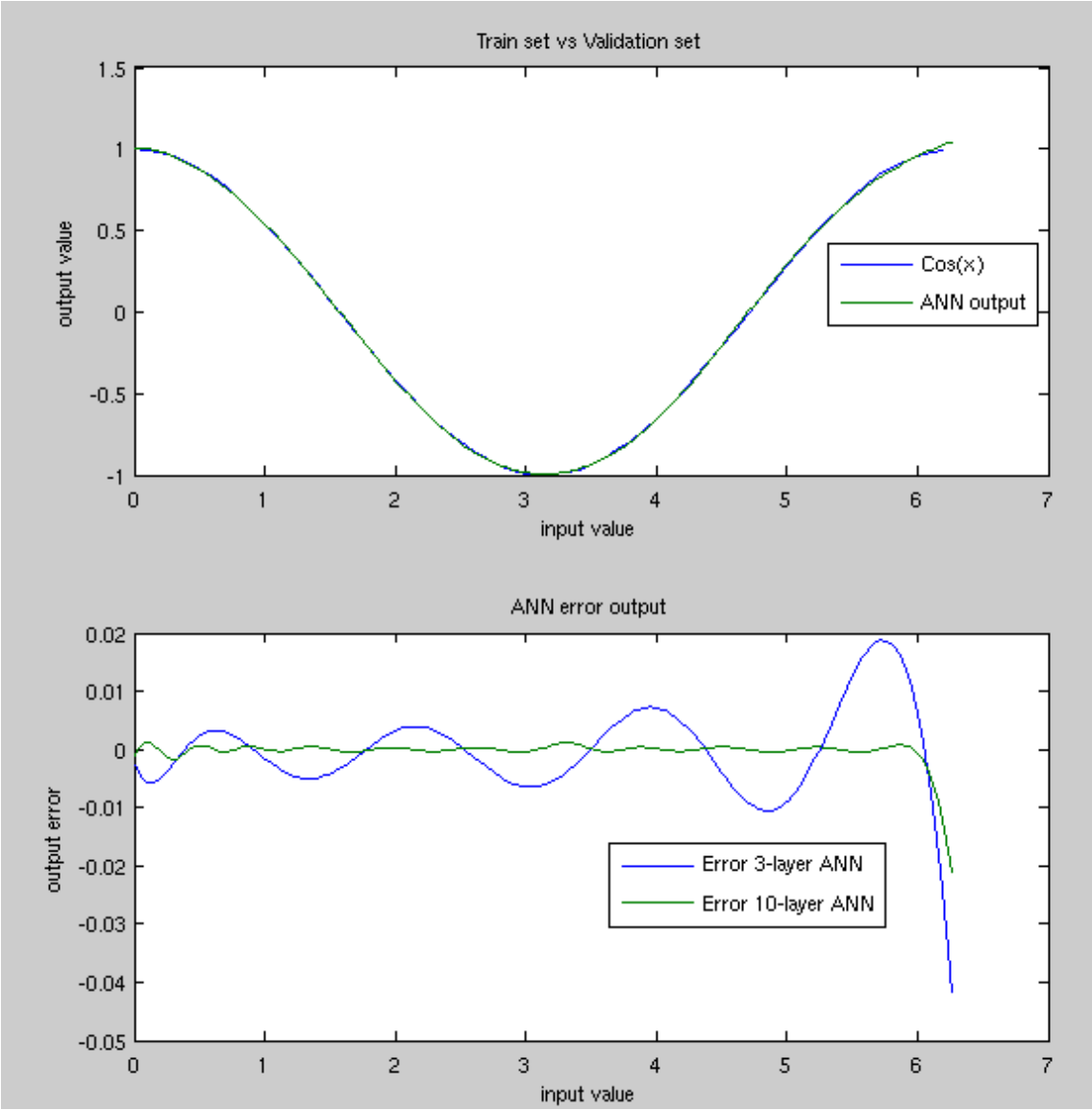
On peut bien voir le travail d'entrainement et le calcul des poids pour minimiser l'erreur. Vérifions maintenant notre système.

3.3 Vérification

Afin de vérifier si notre système fonctionne, on génère maintenant une nouvelle série des valeur de "x" qui n'appartiennent pas à l'ensemble d'entrainement. Voilà le code :

```
x_validation = [0:0.03:2*pi];

y = net(x_validation);
y10 = net10(x_validation);
```

Voilà le résultat (en haut la sortie y, et en bas l'erreur par rapport au vrai cos(x)) :



On voit que l'on à bien pu entrainer notre système avec $x \in [0; 2\pi]$ (au-delà le système est faux, l'erreur grandit). On voit que le système à 10 layers est nettement plus performant que le système à 3 layers. Toutefois, le système devient instable pour des valeurs de x approchant 2π.

Chapitre 4

Train your biometric system

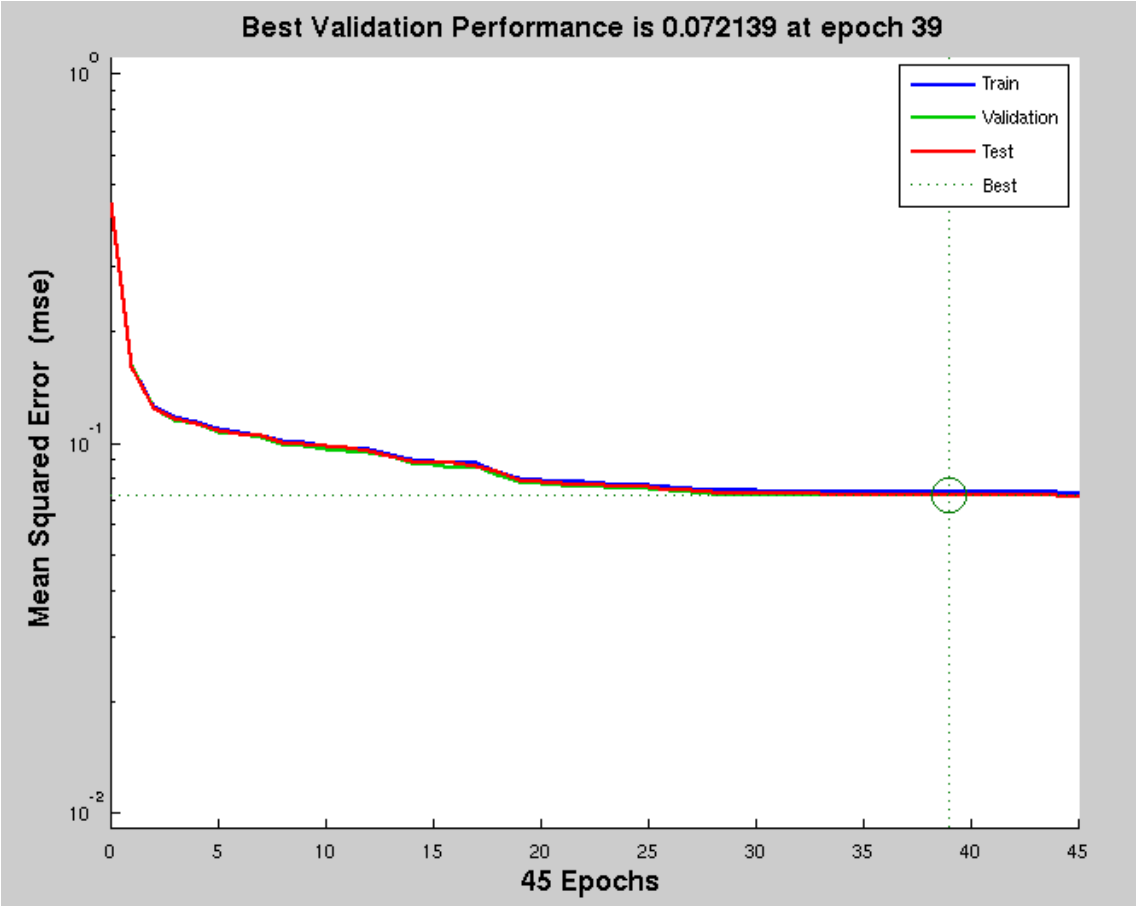
Le but de cet exercice est d'essayer de faire une authentification vocal avec un réseau de neurones. Nous avons enregistré 50 échantillons en fichier "wav". Cet exercice nous a consommé un maximum de temps pour la compréhension de la problématique (ce qui est le plus difficile).

4.1 Feature extraction

Nous avons extrait les différentes caractéristique de nos voix, ainsi que celles de nos collègues. Nous avons simplement pris les deux premières colonnes (5 coefficients retournés par la fonction "FeatureGeneratorTxt"). Une fois les fichiers cvs générés, il faut les relire pour créer les tableau de données matlab.

4.2 ANN training

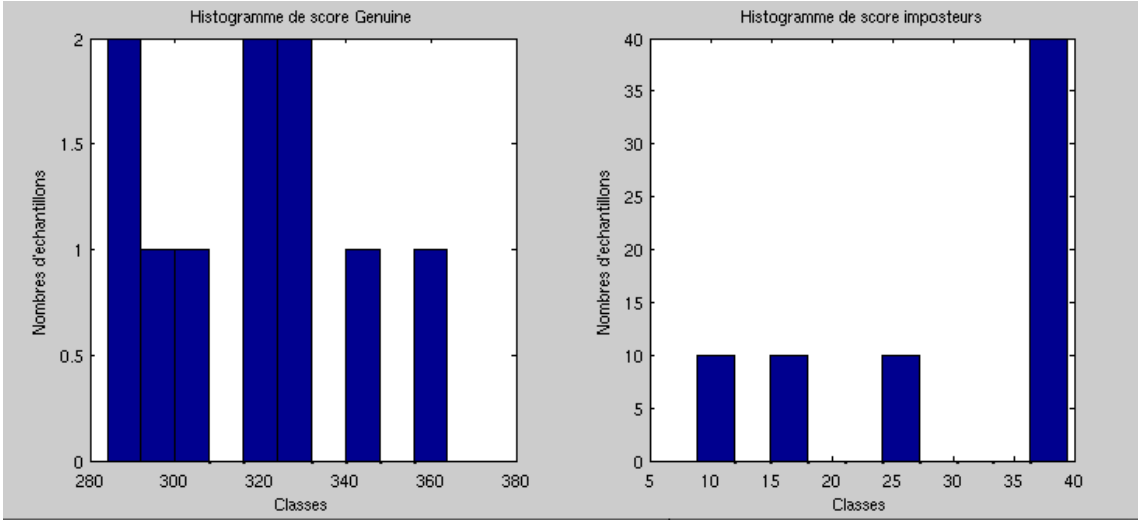
Pour entrainer le système, nous avons pris les premiers 40 fichier "wav" de nos voix et de celles de nos collègues. Le "Target" était "1" si c'était notre voix, et "0" si c'était la voix d'un collègue. Voilà le résultat de l'entrainement :



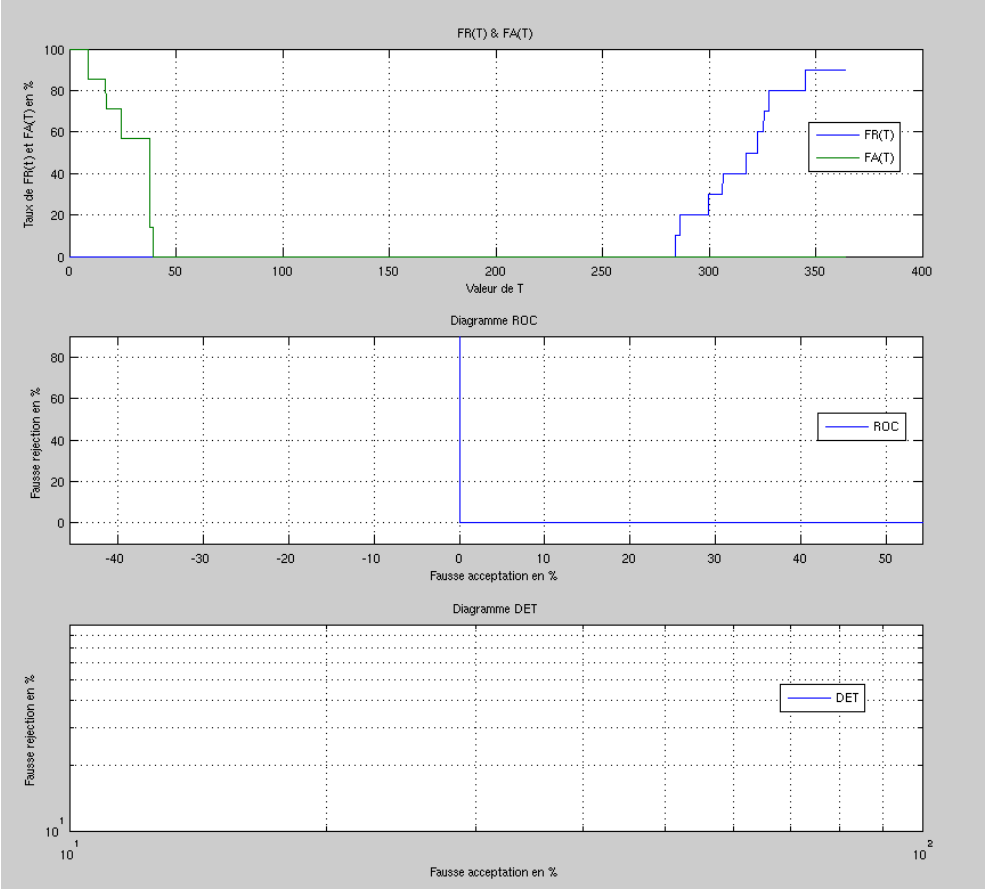
On voit que le système à bien convergé. On a trouvé les meilleures poids possibles pour optimiser l'erreur minimum. Nous pouvons essayer maintenant avec les 10 fichiers restant.

4.3 ANN test

On passe en entrée les 5 colonnes (coefficients extrait) des 10 fichiers "wav" que l'on passe en entrée et on somme les sorties pour calculer les scores. Voilà l'histogramme des scores :



On voit que les scores "genuine" sont entre 250-380 et les "impostors" sont entre 5-40. Voilà les graphiques de performances :



On remarque que les scores ne se recouvrent pas et que nous avons un système qui est parfait !

Chapitre 5

Conclusion

Ce laboratoire nous a permis de mettre en pratique la théorie vue en cours. Les réseaux de neurones sont des outils extrêmement performant pour l'application vue dans ce laboratoire.

Nous aurions pu pousser les recherches un peu plus loin pour avoir de meilleures résultats. Cependant, le temps étant compté, nous n'avons pas réussi à nous investir plus dans ce travail.